

# Cromemco Z80 Monitor

Copyright © 1979 by Cromemco Inc. All rights reserved.



---

# Table of Contents

---

Introduction .....	1
Entry.....	1
System Stack.....	1
Command Format.....	1
Swath Operator.....	1
Multiple Commands: The After Operator.....	1
Example 1: Writing Paper Tapes	
Example 2: Copying Paper Tapes	
Example 3: Multiple Copies of PROMs	
Example 4: Initializing UARTs	
Example 5: A Timer	
Errors and Escapes .....	3
Input and Output .....	3
Baud Rates and UART Selection.....	3
Interrupts .....	3
Installing the Monitor .....	4
Using the Monitor.....	4
COMMANDS.....	4
Display Memory, DM	
Display Registers, DR	
Go, G	
Go with Breakpoints Set, G/	
Initialize Baud Rate, I	
Move, M	
Nulls, N	
Output, O	
Program PROMs, P	
Read, R	
Substitute Memory, SM	
Substitute Register	
SA, SB, SC, SD, SE, SF, SH,	
SI, SN, SP, SS, SX, SY,	
SA', SB', SC', SD', SE', SF', SH'	
UART Select, U	
Verify, V	
Write, W	
Program Listing.....	8
Cross Reference.....	27

## Introduction

The Z80 Monitor makes it possible to control computers which use the CROMEMCO ZPUTM from a terminal keyboard. It includes executive commands to examine and change memory, make a binary or an ASCII dump of memory, move and compare blocks of memory, output a byte of data to any port, read, write, and punch nulls on binary paper tapes, program 2708 and 2704 PROMs using the CROMEMCO BYTESAVER, and initialize and control both serial ports on the CROMEMCO TUART.

Transfer of control to a program in memory can be commanded from the keyboard with up to five breakpoints set and with the initial contents of the ZPU registers specified. When a breakpoint is encountered during execution, control is transferred back to the monitor and the contents of all 22 ZPU registers are stored. These register values can be examined and changed before execution of the program is resumed.

## Entry Points

The Z80 Monitor has three entry points. A cold-start entry at E000 hex selects bank 0 on CROMEMCO memory boards and UART A on the CROMEMCO TUART. It initializes the baud rate of the UART to match that of the terminal being used. In addition, it saves the contents of the Z80 registers I, N (IFF), S (SP), X (IX), Y (IY), A', B', C', D', E', F', and H' (HL') in the user-register area which is part of the system stack. (If the Z80 stack pointer is pointing to RAM, then all registers except A and P (PC) will be saved.) The contents of these registers are restored when the monitor is exited by means of the GO command.

The warm-start entry point at E008 hex is provided so that the monitor can be re-entered without affecting the memory banks or the UART. The same registers are saved as for the cold-start entry point.

The third entry point is used by the breakpoint facility. Entry here saves the contents of all registers. Memory banks and UART are unaffected.

## System Stack

The monitor does not require the user to address a RAM board at a special place in memory for its stack and working storage area. (However, if the breakpoint facility is used, there must be either RAM at locations 30, 31, and 32 hex or PROM with the data C3, 45, E0 hex at those locations.) The monitor finds the highest page of RAM active in the machine and places its stack and temporary storage area there. At least 60H or 96 bytes of this page must be reserved for system use. If the multiple command facility is used,

each additional command in a command line requires an additional 20 hex or 32 bytes stack room. (See Multiple Commands.)

## Command Format

The Z80 Monitor is controlled by one and two-character commands from the terminal keyboard. The format is free-form with respect to spaces.

In the following, DM is the Display Memory command and S is the Swath operator (see below). The four examples are equivalent commands. They display the contents of 100 hex bytes of memory beginning with location 1000 hex. ('(CR)' indicates a carriage return.)

```
DM1000 10FF (CR)  
DM1000S100(CR)  
D M 1000      10FF (CR)  
D M 1000 S 100 (CR)
```

When entering an address as a operand, only the last four digits typed in are retained. For example, '321000' is read as '1000'. Therefore, if a wrong digit is entered, continue typing until the last four digits are correct.

Only the last two digits typed are retained when a two-digit number such as a data byte is entered.

## Swath Operator

There are two ways to specify the address range of many commands. The first is to simply list the beginning and ending addresses (and, where appropriate, the destination address). For example, the first command below programs the range 0 through 13FF into PROMs starting at E400. The second command displays the contents of memory between addresses E400 and E402.

```
PO 13 FF E400  
DME400 E402
```

Another way to do the same thing is to use the Swath operator, S, to specify the width of the address range rather than state the ending address explicitly.

```
PO S1400 E400  
DM E400S3
```

## Multiple Commands: The After Operator

The After operator, '<', can be used to place more than one command on a command line. All of

the commands on the command line are executed before the monitor returns with its prompt ':', for a new command.

With this feature, the monitor can write an area of memory onto paper tape preceded and followed by a sequence of nulls without any undesirable carriage-returns or prompts inserted by the monitor.

### Example 1

Assume that the terminal being used is a teletype-writer with paper tape punch. In order to write the contents of 400 hex bytes starting at 100 hex with a leader of 95 hex nulls and a trailer of 80 nulls, type:

```
:N80 < :W100 S 400 < :N 95 (CR)
```

where the colons are prompts provided by the monitor. Turn on the paper tape punch after typing the carriage-return in order to avoid writing it onto the tape.

There are several points to be made about the use of the After operator:

(a) The order of execution of the commands is from right to left. Hence, the name 'After' and the shape '<'.

(b) The After operator is logically equivalent to a carriage-return. Anywhere a carriage-return can reasonably appear in a command, the After operator may be used instead. However, no commands in the line are executed until an actual carriage-return is typed.

(c) If any of the GO commands appears in a multiple-command line, it must be the last command executed, i.e., the first command typed.

(d) Each additional command on a line adds from 10 to 20 hex bytes to the system stack size.

### Example 2

Assume that we are using a CROMEMCO TUART I/O card with a console connected to UART A and with a paper tape reader and punch connected to the input and output, respectively, of UART B. Assume that the baud rate of UART B has already been set to that of the reader and punch. (See Baud Rates pg. 3.) We can copy a paper tape by switching the current UART to B, reading the tape into a memory buffer, writing a leader, writing the buffer to the punch, and finally switching the current UART back to A, the console, by typing:

```
:UA < :W0S2000 < :N80 < :R0S2000 < :UB (CR)
```

In this case, we can leave the reader and punch on all the time. There is no question of a carriage-return from the command line being punched onto the paper tape since two different UARTs are involved.

Perhaps we forgot to write nulls as a trailer to the output tape. After the prompt, ':', again appears on the console, we can rectify this by typing:

```
:U A < :N 80 < :U B (CR)
```

where, again, all colons are provided by the monitor.

### Example 3

Suppose we wish to make three copies of the same PROM. Assume that the source is in RAM at location 0 and that we want three identical copies in PROMs located at E400, E800, and EC00 hex. The following command line will accomplish this:

```
:POS400 EC00 < :P0S400 E800 < :POS400 E400 (CR)
```

### Example 4

Either of the following will initialize the baud rate of a terminal connected to UART B of the TUART:

```
:I < :UB (CR)  
:UA < :I < :UB (CR)
```

After entering one of these commands on the console connected to UART A, push CARRIAGE-RETURN on the other terminal until the monitor prompt ':' appears.

### Example 5

Assume that we would like to take a brief nap to refresh ourselves but have no alarm clock. Assume further that two beeps of the console bell spaced 2.1 seconds apart are sufficient to wake us and that the console can run at 300 baud. Since the Display Memory command takes 63 characters to display 10 hex or 16 bytes of memory, at 300 baud it takes 2.1 seconds or 0.035 minutes to display 10 hex bytes.

Number of Bytes (hex)	Time (minutes)
10	0.035
640	3.5
C80	7.0
1900	14.0
3200	28.0
6400	56.0
C800	112.0

First, we re-initialize the UART by typing the following:

```
:I (CR)
```

Set the console baud rate to 300 and push the CARRIAGE-RETURN until the monitor issues its prompt, ':'.

To ring the bell, output 7 to port 1. For a nap of 14 minutes:

```
:O 7 1 < :DM0S10 < :O7 1 < :DM0S1900 (CR)
```

## Errors and Escapes

When the monitor detects an error condition, the command is aborted, all breakpoints are cleared, and a '?' is printed followed by the prompt ':' for the next command.

Any command may be aborted from the keyboard either when the monitor is requesting further input, or during print-out, by depressing either the ESCAPE or the ALT MODE key. CONTROL-SEMI-COLON, CONTROL-SHIFT-'K', and '}' may also work.

## Input and Output

The monitor assumes that a data transfer occurs on I/O port 1. Status flags are transmitted over input port 0. The data-available flag is on bit 6 of input port 0. The transmitter-buffer-empty flag is on bit 7 of input port 0. Both flags are active high.

To use the CROMEMCO TUART with the monitor, set switches 1, 7, and 9 of the 10-position TUART switch OFF, all others ON. The currently selected UART uses I/O port 1 for date transfer and input port 0 for status flags. The UART which is not current uses I/O port 51 hex for date transfer and input port 50 hex for status flags. (The UARTs are selected by means of the UART command.)

The following locations may be changed for different I/O conventions:

Status port number (00): E00F, E020  
Input data port number (01): E014  
Output data port number (01): E027  
Input-data-available mask (40): E011  
Output-transmitter-buffer-empty mask (80): E022

For active-low status flags change locations E019 and E379 from 28 hex to 20 hex and change location E120 from 20 hex to 28 hex.

## Baud Rates and UART Selection

When the monitor is entered at E000 hex, the cold-start entry point, push CARRIAGE-RETURN until the monitor responds with:

CROMEMCO ZM1.4

The monitor is capable of selecting 19200, 9600, 4800, 2400, 1200, 300, 150, or 110 baud when used with the CROMEMCO TUART I/O board.

The maximum number of carriage-returns required to select any of these baud rates is four. (Two carriage-returns are required for any UART with a fixed baud rate.)

The baud rate can also be changed by using the Initialize command (see page 5).

Some peripheral devices such as paper tape readers or punches may have no keyboards. The TUART baud rate can also be set by outputting a data byte from the following table to port 0 for the currently selected UART or to port 50 hex for the unselected UART. (To make UART B current, output 80 hex to port 4. For UART A, output 0 to port 54 hex. UART selection can also be accomplished by means of the monitor's UART command, U).

Baud Rate	Data Byte
110	01
150	82
300	84
1200	88
2400	90
4800	A0
9600	C0

The baud rate can be octupled by outputting 10 hex to port 2 for the selected UART or to port 52 hex for the other UART. Outputting 0 to these ports brings the baud rate back to normal.

## Interrupts

The monitor can be used to enable interrupts in the Z80. This is done by changing the value of the N register to 1 by using the Substitute Register command, SN. (The N register stores the value of the Z80 interrupt flip-flop at the time the monitor is entered.) Then interrupts will be enabled when one of the Go commands is given.

Note, however, that the interrupt mask registers on the TUART must have been set previously, either by a user program or by the monitor. (If this is not done, then an immediate interrupt will be generated because the print buffer is empty.) To mask out all interrupts output 0 to port 3 for the current UART and to port 53 hex for the other UART.

The mask bit corresponding to each of the possible interrupts is given in the following table:

Bit	Interrupting Device
0	Timer 1
1	Timer 2
2	Sens (external)
3	Timer 3
4	Receiver Data Available
5	Transmitter Buffer Empty
6	Timer 4
7	Timer 5 or external

or

DM beginning-addr S swath-width (CR)

The contents of memory are displayed in hexadecimal form. Each line of the display is preceded by the address of its first byte. Example:

:DM100 S3  
0100: C3 34 7F

## DISPLAY REGISTERS

### [2] DR (CR)

When the monitor is re-entered from a breakpoint, the contents of all the Z80 registers are stored in an area called the user-register area. (When the monitor is entered via reset or the warm-start entry point, all registers except A, B, C, D, E, F, HL, and P are saved in the user-register area. However, if the stack pointer is pointing to RAM, then all but A and P will be saved.)

DR causes these stored registers to be displayed in the following format:

A=01 B=12 C=34 D=56 E=78 F=9A HL=BCDE  
I=F0 N=00 P=1234 S=5678 X=9ABC Y=DEF0  
A '23 B '45 C '67 D '89 E 'AB F 'CD HL 'EF01

If interrupts were enabled when the monitor was entered, then N=1. Otherwise, N=0.

The flag registers, F and F', are packed as follows:

S,Z,x,H x,P/V,N,C

i.e., sign, zero, (unknown), half-carry, (unknown), parity or overflow, subtraction, and carry flags.

### GO

### [3] G (CR)

The Z80 registers are loaded with the values saved in the user-register area. (These are the values displayed with the DR command.) Execution then resumes at the location contained in the user-program-counter, P.

### [4] G starting-addr (CR)

This command is exactly like [3] except that the user-program-counter, P, is first loaded with starting-address. Thus, execution begins at starting-address.

## GO WITH BREAKPOINTS SET

### [5] G / breakpoint-addr-1 breakpoint-addr-2 . . . (CR)

[6] G starting-addr /brkpt-addr-1 brkpt-addr-2 . . . (CR)

Commands [5] and [6] are like [3] and [4], respectively, except that breakpoints are set at breakpoint-address-1, breakpoint-address-2, etc.

When a breakpoint is encountered in the execution of the user program, the monitor is re-entered. All registers are saved in the user register area (which is part of the system stack), the address of the breakpoint is printed, and all breakpoints are cleared (i.e., the user program is restored to its original state). Finally, the prompt, ‘:’ is issued for the next command from the keyboard. Note the following about the use of breakpoints:

(a) Breakpoints can only be set in programs residing in RAM. This is because the monitor inserts a RST 48 instruction (F7 hex) at each breakpoint location. (The original contents of these locations are saved so that they can later be restored.)

(b) Up to five breakpoints can be set. If an attempt is made to set a sixth breakpoint, the monitor will print a question mark to indicate error, erase all breakpoints, and prompt for a new command.

(c) When a breakpoint is set, the monitor inserts a 3-byte jump instruction at location 30 hex. This means that locations 30, 31, and 32 hex are not available to the user program when breakpoints are used.

(d) The monitor temporarily uses ten bytes on the user's stack in executing a breakpoint. The area reserved for the user's stack must, therefore, be at least ten bytes larger than that required for the user's program.

(e) If breakpoints are set in a program and the computer is reset and the monitor re-entered before any breakpoint is reached in the execution of the program, then the breakpoints will have to be removed from the program by means of the Substitute Memory command, SM. However, if any breakpoint is reached, all breakpoints are automatically cleared by the monitor.

## INITIALIZE BAUD RATE

[7] I (CR)

After the CARRIAGE-RETURN is typed, change the baud rate of the terminal to the desired value and then push the CARRIAGE-RETURN until the monitor responds with its prompt, ‘:’.

The monitor is capable of selecting 19200, 9600, 4800, 2400, 1200, 300, 150, or 110 baud when used with the Cromemco TUART I/O board. The maximum number of carriage-returns required to select any of these baud rates is four.

The command is particularly useful for setting the baud rate of the second serial port on the TUART. (See Multiple Commands.)

## MOVE

[8] M source-addr source-end destination-addr (CR)

or

M source-addr S swath-width destination-addr (CR)

Move the contents of memory beginning with source-address and ending with source-end to destination-address. After the move, the monitor verifies that source and destination are the same. This will result in a print-out of discrepancies which are not really errors after certain types of overlapping moves. However, this print-out can be terminated by depressing ESCAPE or ALT MODE.

The Move command can be used to fill a block of memory with a constant. For example, to enter zeros between locations 100 and 108, use the Substitute Memory command to enter 0 at location 100, and then move 100 through 107 to 101:

M100 107 101

or

M 100 S 8 101

Care should be taken not to overwrite the system stack which resides in the top of active RAM. (See System Stack.)

## NULLS

[9] N hex-number (CR)

Write hex-number nulls to the current device. This command is used to punch leaders and trailers on paper tape. (See Multiple Commands.)

## OUTPUT

[10] O data-byte port-number (CR)

Outputs data to a port. One use of this command is to select banks on Cromemco memory boards. When the monitor is first entered on power-up or reset, it selects bank 0 and turns off all other memory banks.

Either a software output or a monitor output to port 40 hex serves to change the bank selection. To select bank n, output a byte with bit n high. To select two banks, n and m, output a byte with both bits n and m high.

Bank	Output byte
0	01
1	02
2	04
3	08
4	10
5	20
6	40
7	80

For example, the first command selects bank 5 and the second selects banks 4 and 5.

O 20 40  
O 30 40

## PROGRAM

[11] P source-addr source-end destination-addr (CR)

or

P source-addr S swath-width destination-addr (CR)

Program from source-address through source-end into PROMS beginning at destination-address.

If the length of the source is not a multiple of 400H (1024 decimal) or if the destination does not begin at 400H boundary, the monitor will reject the command. (Multiples of 400H end in '000', '400', '800', or 'C00'.)

Any number of 2708 or 2704 PROMS can be programmed in the execution of one command as long as there are enough BYTESAVERS to contain them. Each PROM is verified with its source after all are programmed and any discrepancies are printed out. If there are none, the prompt ':' is issued and the monitor awaits the next command.

Software can be loaded into a PROM in as small increments as you desire provided it is added to previously unused areas of the PROM.

This is done by first using the Move command, M, to transfer the current contents of the PROM down to RAM, adding the new software to an area of RAM which corresponds to the unused portion of the PROM and finally using the Program command, P, to re-program the PROM with the result.

Although the entire PROM must always be programmed, it never hurts to re-write the same data over again.

In general, a 1 may be written over a 1, a 0 over either a 1 or a 0, but the only way to change 0's to 1's is to erase the PROM with appropriate UV light. (See the BYTESAVER manual for details.)

## READ

[12] R destination-addr destination-end (CR)

or

R destination-addr S swath-width (CR)

Read binary or ASCII input from paper tape reader or console and store in memory from destination-address through destination-end. After destination-end has been filled, the monitor prompts for the next command.

## SUBSTITUTE MEMORY

[13] SM address (CR)

Substitute Memory displays the contents of address and outputs a dot, '.', as a prompt for the substituted value. If no change is desired, type a space or another dot. Otherwise, enter the new value. The monitor accepts hex digits until it gets a delimiter, such as a space, dot, or carriage-return retaining the last two digits entered as the value. Unless the delimiter is a carriage-return, the monitor outputs the contents of the next sequential memory location with a dot prompt. A carriage-return terminates the command.

## SUBSTITUTE REGISTER

[14] S register-name (CR)

Register-name may be A, B, C, D, E, F, H (HL), I, N (state of the Z80 interrupt flip-flop), P (PC), S (SP), A', B', C', D', E', F', H' (HL'), X (IX), or Y (IY).

This command prints the name of the user-register requested, displays its contents, outputs a dot, '.', as a prompt for the substituted value. If no change is desired, type a space or another dot. Otherwise, enter the new value. The monitor accepts hex digits until it gets a delimiter such as space, dot, or carriage-return retaining the last two digits (four digits for a 2-byte register). Unless the delimiter is a carriage-return, the monitor prints the name and contents of the next register followed by the dot prompt. A carriage return terminates the command.

## UART SELECT

[15] U device-name (CR)

Device-name may be A or B. The Cromemco TUART has two UARTs. When the monitor is entered via reset, UART A is selected for its input/output channel. This command allows the user to change the UART selection. It is often used in the multiple command mode (see page 2).

## VERIFY

[16] V source-addr source-end destination-addr (CR)

or

V source-addr S swath-width destination-addr (CR)

Verify that the block of memory between source-address and source-end contains the same values as the block beginning at destination-address. The addresses and contents are printed for each discrepancy found (unless the print-out is terminated by ESCAPE or ALT MODE).

This command works by reading bytes from the source and destination and comparing them. If a discrepancy is found, the memory is read again for print-out. Thus, it can happen that a discrepancy is printed-out with the source and destination contents indicated to be the same. This is caused by a defective memory element.

## WRITE

[17] W source-addr source-end (CR)

or

W source-addr S swath-width (CR)

Write binary or ASCII output from source-address through source-end to the current device (selected by the UART command). After source-end has been written, the monitor prompts for the next command.

The Write command is useful for punching binary or ASCII paper tapes of the contents of memory and for looking at the ASCII contents of memory on the console.

When punching a paper tape, it is usually desirable to punch series of nulls as leader and trailer. This can best be done in conjunction with the Null command and the After operator. (See Multiple Commands for examples of this usage.)

---

## Program Listing

---

```

        0002 ;
        0003 ;
(0000) 0004 STAT:   EQU    0      ;STATUS PORT, DEVICE A
(0001) 0005 DATA:   EQU    1      ;DATA PORT, DEVICE A
(0002) 0006 ACMNDP: EQU    2      ;COMMAND PORT, DEV. A
(0000) 0007 ABAUDP: EQU    0      ;BAUD PORT, DEVICE A
(0004) 0008 APARLP: EQU    4      ;PARALLEL PORT, DEV. A
(0052) 0009 BCMNDP: EQU 52H    ;COMMAND PORT, DEV. B
(0054) 0010 BPARLP: EQU 54H    ;PARALLEL PORT, DEV. B
(0040) 0011 DAV:    EQU 40H    ;DATA-AVAILABLE MASK
(0080) 0012 TBE:    EQU 80H    ;XMITTER-BUF-EMPTY MSK
        0013 ;
(0005) 0014 NBRKPT: EQU    5      ;ALLOW ROOM FOR
(0016) 0015 BPSTOR: EQU NBRKPT*4+2 ;BREAKPOINT STORAGE
(0016) 0016 TEMPS:  EQU BPSTOR
(000B) 0017 BPMRK: EQU 0BH    ;USED TO MARK THE SET-
        0018 ;TING OF A BP IN BPSTOR.
(0030) 0019 RSTLC:  EQU 30H    ;RST LOCATION
(0000) 0020 CASE:  EQU 0      ;(REQUIRES UPPER-CASE)
(0005) 0021 B2F:   EQU 5      ;2-BYTE FLAG
(0006) 0022 PF:    EQU 6      ;PRIME-ABLE REG FLAG
(0007) 0023 CRF:   EQU 7      ;CRLF FLAG
        0024 ;
(000D) 0025 CR:    EQU 0DH
(000A) 0026 LF:    EQU 0AH
(001B) 0027 ESC:   EQU 1BH
(007D) 0028 ALT:   EQU 7DH
        0029 ;
0030 ; DISPLACEMENTS FROM IX OF HI BYTE OF REG PAIRS
0031 ;
0032 ;
(FFFF) 0033 DUPC:  EQU -1
(FFFD) 0034 DUAF:  EQU -3
(FFFB) 0035 DUBC:  EQU -5
(FFF9) 0036 DUDE:  EQU -7
(FFF7) 0037 DUHL:  EQU -9
(FFF5) 0038 DUSP:  EQU -11
(FFF3) 0039 DUIX:  EQU -13
(FFF1) 0040 DUIY:  EQU -15
(FFEF) 0041 DUIN:  EQU -17      ;I & THE INTERRUPT FF
(FFED) 0042 DUAF2: EQU -19
(FFEB) 0043 DUBC2: EQU -21
(FFE9) 0044 DUDE2: EQU -23
(FFE7) 0045 DUHL2: EQU -25
        0046 ;
(001A) 0047 LENRGS: EQU DUPC-DUHL2+2
        0048 ;
        0049 ;
        0050 ;
        0051 ;
E000          0052         ORG    0E000H
        0053 ;
0054 ; ENTER THE MONITOR FROM RESET.
0055 ; COLD START ENTRY. INITIALIZES THE UART
0056 ; AND ZEROES THE BREAKPOINT STACK POINTER.
0057 ; ALTERS THE A-REGISTER. SAVES ALL OTHER
0058 ; REGISTERS EXCEPT THE PROGRAM COUNTER,

```

```
0059 ; BUT DOES NOT DISPLAY THEM.  
0060 ;  
E000 3E01 0061 CSTART: LD A,1  
E002 D340 0062 OUT 40H,A ;SELECT BANK 0  
E004 F5 0063 PUSH AF ;SIMULATE UPC  
E005 F5 0064 PUSH AF ;USER-F-REGISTER  
E006 1842 0065 JR COMMON  
0066 ;  
0067 ;  
0068 ;  
0069 ; WARM START ENTRY. INITIALIZES THE BREAKPOINT  
0070 ; STORAGE POINTER. SAVES ALL REGISTERS EXCEPT  
0071 ; THE PROGRAM COUNTER, BUT DOES NOT DISPLAY THEM.  
0072 ;  
E008 F5 0073 WSTART: PUSH AF ;SIMULATE UPC  
E009 F5 0074 PUSH AF ;UAF  
E00A 3E80 0075 LD A,80H ;FLAG:  
E00C 183C 0076 JR COMMON ;WARM-START ENTRY  
0077 ;  
0078 ;  
0079 ; CHECK INPUT & RETURN WITH DATA IF READY.  
0080 ;  
E00E DB00 0081 CHKIN: IN A,STAT  
E010 E640 0082 AND DAV  
E012 C8 0083 RET Z  
E013 DB01 0084 IN A,DATA  
E015 C9 0085 RET  
0086 ;  
0087 ;  
0088 ; GET CHARACTER FROM INPUT.  
0089 ;  
E016 CD0EE0 0090 GBYTE: CALL CHKIN  
E019 28FB 0091 JR Z,GBYTE  
E01B E67F 0092 AND 7FH  
E01D C9 0093 RET  
0094 ;  
0095 ;  
0096 ; PRINT CHARACTER.  
0097 ;  
E01E F5 0098 PBYTE: PUSH AF  
E01F DB00 0099 PBY1: IN A,STAT  
E021 E680 0100 AND TBE  
E023 28FA 0101 JR Z,PBY1  
E025 F1 0102 POP AF  
E026 D301 0103 OUT DATA,A  
E028 C9 0104 RET  
0105 ;  
0106 ;  
0107 ; SELECT DEVICE A & INITIALIZE ITS BAUD RATE.  
0108 ; ENTER WITH A=1.  
0109 ;  
E029 D354 0110 INIT: OUT BPARLP,A ;SELECT DEVICE A  
E02B D352 0111 OUT BCMNDP,A ;RESET DEVICE B  
0112 ; ;[CONTINUE BELOW]  
0113 ;  
0114 ;  
0115 ; INITIALIZE BAUD RATE OF THE CURRENT DEVICE.
```

```
0116 ;
0117 ; PUSH CARRIAGE-RETURN TO SELECT THE PROPER BAUD
0118 ; RATE FOR THE CURRENT TERMINAL. (THE MAXIMUM
0119 ; NUMBER OF CARRIAGE-RETURNS REQUIRED IS FOUR.)
0120 ;
0121 ; WITH THE CROMEMCO TUART ANY OF THE FOLLOWING
0122 ; BAUD RATES CAN BE SELECTED:
0123 ; 19200, 9600, 4800, 2400, 1200, 300, 150, 110.
0124 ;
0125 ; WITH THE 3P+S: 2400, 300, 110.
0126 ;
0127 ; TWO CARRIAGE-RETURNS ARE REQUIRED FOR
0128 ; ANY UART WITH A FIXED BAUD RATE.
0129 ;
E02D 21A3E3 0130 INITBAUD: LD     HL,BAUDRS
E030 0E00    0131      LD     C,ABAUDP
E032 3E11    0132      LD     A,11H          ;OCTUPLE THE CLOCK
E034 D302    0133 IT1:    OUT   ACMNDP,A      ;& RESET CURRENT DEVICE
E036 EDA3    0134      OUTI
E038 CD16E0  0135      CALL  GBYTE
E03B CD16E0  0136      CALL  GBYTE
E03E FE0D    0137      CP    CR
E040 3E01    0138      LD    A,1          ;SLOW THE CLOCK
E042 20F0    0139      JR    NZ,IT1
E044 C9      0140      RET
0141 ;
0142 ;
0143 ; BREAKPOINT ENTRY. INITIALIZES NOTHING.
0144 ; SAVES ALL REGISTERS AND DISPLAYS THEM.
0145 ;
E045 E3      0146 SVMS:   EX    (SP),HL        ;ADJUST BRKPT
E046 2B      0147 DEC     HL               ;RET ADDR
E047 E3      0148 EX     (SP),HL
E048 F5      0149 PUSH    AF               ;UAF
E049 97      0150 SUB     A                ;FLAG:
0151 ;           ;BREAKPOINT ENTRY;
0152 ;
0153 ;
E04A C5      0154 COMMON: PUSH   BC               ;UBC
E04B 47      0155 LD      B,A               ;ENTRY FLAG
E04C D5      0156 PUSH   DE               ;UDE
E04D E5      0157 PUSH   HL               ;UHL
0158 ;
0159 ; PLACE SYS STACK AT HIGHEST PAGE OF
0160 ; AVAILABLE RAM.
0161 ; ALLOW ROOM FOR TEMP STORAGE.
0162 ;
E04E 21E900  0163 LD     HL,00FFH-TEMPS
E051 25      0164 COM1:  DEC    H
E052 7E      0165 LD     A,(HL)
E053 34      0166 INC    (HL)
E054 BE      0167 CP    (HL)          ;DID IT CHANGE?
E055 28FA  0168 JR    Z,COM1
E057 35      0169 DEC    (HL)          ;YES. RESTORE IT.
0170 ;
E058 78      0171 LD     A,B               ;ENTRY FLAG
E059 EB      0172 EX     DE,HL
```

```

E05A 210900      0173      LD      HL,9
E05D 39          0174      ADD     HL,SP      ; -> UPC, HI BYTE
E05E 010A00      0175      LD      BC,10
E061 EDB8        0176      LDDR
E063 13          0177      ;
E064 EB          0178      INC     DE      ;-> UHL,LO ON SYS STK
E065 F9          0179      EX      DE,HL
E066 EB          0180      LD      SP,HL      ;CURRENT SYS SP
E067 010B00      0181      EX      DE,HL
E06A 09          0182      LD      BC,DUPC-DUHL+3
E06B E5          0183      ADD     HL,BC      ;HL = USER SP
E06C DDE5        0184      PUSH    HL      ;USP
E06E FDE5        0185      PUSH    IX      ;UIX
E06F FDE5        0186      PUSH    IY      ;UIY
E070 EB          0187      EX      DE,HL
E071 09          0188      ADD     HL,BC
E072 4D          0189      LD      C,L      ;SAVE
E073 2B          0190      DEC     HL
E074 E5          0191      PUSH    HL
E075 DDE1        0192      POP     IX
E077 FE01        0193      CP      I`      ;ENTRY?
E079 3807        0194      JR      C,COM3      ;SKIP IF VIA BP.
E07B 71          0195      LD      (HL),C      ;BP PNTR, LO BYTE
E07C 23          0196      INC     HL
E07D 3600        0197      LD      (HL),0      ;BP-STACK ENDMARK
E07E 0198        ; INITIALIZE THE TUART IF ENTRY WAS VIA RESET.
E07F CC29E0        0199      ; (A CONTAINS 1.)
E07F CC29E0        0200      ;
E082 ED57        0201      CALL    Z,INIT
E084 67          0202      ;
E085 2E00        0203      COM3:   LD      A,I
E087 E28BE0      0204      LD      H,A
E08A 2C          0205      LD      L,0
E08B E5          0206      JP      PO,COM4
E08A 2C          0207      INC     L
E08B E5          0208      COM4:   PUSH    HL      ;UIN
E08C 08          0209      EX      AF,AF'
E08D F5          0210      PUSH    AF      ;UAF'
E08E 08          0211      EX      AF,AF'
E08F D9          0212      EXX
E090 C5          0213      PUSH    BC      ;UBC'
E091 D5          0214      PUSH    DE      ;UDE'
E092 E5          0215      PUSH    HL      ;UHL'
E093 D9          0216      EXX
E094 21F0E3        0217      ;
E097 D40FE2        0218      ; IF CY IS SET, ENTRY WAS VIA A BREAKPOINT
E09A 018650        0219      LD      HL,HEAD
E09D DC23E3        0220      CALL    NC,PMSG
E09A 018650        0221      LD      BC,[['P'+CASE] SHL 8]+86H ;IF BP ENTRY,
E09D DC23E3        0222      CALL    C,SUBR3      ;DISPLAY THE PC.
E094 21F0E3        0223      ;
E097 D40FE2        0224      ;
E09A 018650        0225      ;CLEAR ALL BREAKPOINTS
E09D DC23E3        0226      ;
E094 21F0E3        0227      ;
E0A0 DDE5          0228      CLBP:   PUSH    IX
E0A2 E1            0229      POP     HL      ;POINTS TO BPSP,LO

```

E0A3	6E	0230	LD	L, (HL)	; BPSP NOW IN HL
		0231	;		
E0A4	7E	0232	CL1:	LD A, (HL)	; BP STK EMPTY?
E0A5	FE0B	0233	CP BPMRK		; IF BPMRK, BP IS SET
E0A7	200A	0234	JR NZ, CL2		
		0235	;		
E0A9	34	0236	INC (HL)		; BP-ERASED MARK
E0AA	2B	0237	DEC HL		
E0AB	56	0238	LD D, (HL)		
E0AC	2B	0239	DEC HL		
E0AD	5E	0240	LD E, (HL)		
E0AE	2B	0241	DEC HL		
E0AF	EDA8	0242	LD LDD		; RESTORE MEM CONTENTS
E0B1	18F1	0243	JR CL1		
		0244	;		
E0B3	7D	0245	CL2:	LD A, L	
E0B4	2B	0246	DEC HL		
E0B5	77	0247	LD (HL), A		; ADJUST BPSP
		0248	;		
E0B6	11E6FF	0249	LD DE, -LENRGS		; FOR THE BENEFIT
E0B9	19	0250	ADD HL, DE		; OF ERROR & ESCPE
E0BA	F9	0251	LD SP, HL		; RE-INITIALIZE SP
		0252	;		
		0253	;		
		0254	; GET 1-BYTE COMMAND.		
		0255	; RETURNS VALUE IN HL & JUMPS TO THAT ADDR.		
		0256	;		
E0BB	CD4DE1	0257	CALL CRLF		
E0BE	11BEE0	0258	CMND: LD DE, CMND		; SET-UP RETURN
E0C1	D5	0259	PUSH DE		
E0C2	21AEE3	0260	CMND1: LD HL, PRMPT		; RE-ENTRY POINT
E0C5	CD0FE2	0261	CALL PMSG		; FOR RECURSION
		0262	; HL NOW PNTS TO THE COMMAND TABLE.		
		0263	;		
		0264	; GET THE COMMAND.		
		0265	; DE GETS THE FIRST ALPHA CHAR LESS 'D'.		
		0266	;		
E0C8	CDDDEL	0267	CALL SKSGØ		; GET NON-SPACE
E0CB	C8	0268	RET Z		; IF CR, IGNORE.
E0CC	D644	0269	SUB 'D'+CASE		; < 'D'?
E0CE	3815	0270	JR C, ERROR		
E0D0	FE14	0271	CP 'W'-'D'+1		; > 'W'?
E0D2	3011	0272	JR NC, ERROR		
E0D4	5F	0273	LD E, A		
E0D5	1600	0274	LD D, Ø		
		0275	;		
E0D7	4A	0276	LD C, D		; INITIALIZE FOR SUBR
E0D8	EB	0277	EX DE, HL		
E0D9	29	0278	ADD HL, HL		; TIMES 2
E0DA	19	0279	ADD HL, DE		; + TBL ADDR
E0DB	5E	0280	LD E, (HL)		
E0DC	23	0281	INC HL		
E0DD	56	0282	LD D, (HL)		
E0DE	EB	0283	EX DE, HL		
E0DF	CDDDEL	0284	CALL SKSGØ		; NEXT CMND GHAR
E0E2	FE4D	0285	CP 'M'+CASE		; (USED IN SUBST & DISPL)
E0E4	E9	0286	JP (HL)		

```
0287 ;
0288 ;
0289 ; ERROR & ESCAPE. RETURNS TO CMND WITH SP
0290 ; POINTING TO SAVED-REG AREA (UHL').
0291 ;
E0E5 3E3F 0292 ERROR: LD A,'?'
E0E7 CD12E1 0293 CALL PCHR
E0EA 18B4 0294 ESCPE: JR CLBP ;CLEAR ANY BRKPTS
0295 ;
0296 ;
0297 ; PROGRAM PROMS. ABORTS IF DESTINATION
0298 ; IS NOT ON A 1K (400H) BOUNDARY, OR IF SWATH
0299 ; WIDTH IS NOT A MULTIPLE OF 1K.
0300 ;
0301 ;
E0EC CDA5E1 0302 PROG: CALL L3NCR
E0EF 78 0303 LD A,B ;ARE INCREMENT &
E0F0 B2 0304 OR D ;DESTINATION BOTH
E0F1 E603 0305 AND 3 ;MULTIPLES OF
E0F3 B1 0306 OR C ;1024?
E0F4 B3 0307 OR E
E0F5 20EE 0308 ERRV1: JR NZ,ERROR ;ERROR VECTOR
0309 ;
E0F7 E5 0310 PUSH HL ;SOURCE
E0F8 214001 0311 LD HL,320 ;# OF ITERATIONS
E0FB E3 0312 PR1: EX (SP),HL
E0FC CD1AE2 0313 CALL MVE ;MOVE IT
E0FF E3 0314 EX (SP),HL
E100 2B 0315 DEC HL ;ITERATION CT
E101 7C 0316 LD A,H
E102 B5 0317 OR L
E103 20F6 0318 JR NZ,PR1
E105 E1 0319 POP HL
E106 1861 0320 JR VRFY ;VERIFY IT
0321 ;
0322 ;
0323 ; PRINT THE 2 BYTES IN (HL) & (HL-1).
0324 ; DECREMENTS HL BY 2. ALTERS A.
0325 ; PRESERVES OTHER REGS.
0326 ;
E108 CDECE1 0327 P2NMS: CALL PNM
E10B 2B 0328 DEC HL
E10C CDECE1 0329 CALL PNM
E10F 2B 0330 DEC HL ;(CONTINUE BELOW)
0331 ;
0332 ;
0333 ; PRINT SPACE. ALTERS A.
0334 ;
E110 3E20 0335 SPACE: LD A,20H ;(CONTINUE BELOW)
0336 ;
0337 ;
0338 ; PRINT THE CHARACTER IN THE A-REGISTER.
0339 ; (CHKS INPUT FOR ESC.) PRESERVES ALL REGS.
0340 ;
E112 F5 0341 PCHR: PUSH AF ;SAVE THE CHAR
E113 E67F 0342 PCl: AND 7FH
E115 FE1B 0343 CP ESC
```

E117	28D1	0344	JR	Z,ESCAPE	
E119	FE7D	0345	CP	ALT	; ALT MODE?
E11B	28CD	0346	JR	Z,ESCAPE	
E11D	CD0EE0	0347	CALL	CHKIN	
E120	20F1	0348	JR	NZ,PC1	
		0349 ;			
E122	F1	0350 PC2:	POP	AF	
E123	E5	0351	PUSH	HL	
E124	F5	0352	PUSH	AF	
E125	E67F	0353	AND	7FH	
E127	CD1EE0	0354	CALL	PBYTE	
E12A	21ABE3	0355	LD	HL,LFNN	
E12D	FE0D	0356	CP	CR	
E12F	CC0FE2	0357	CALL	Z,PMSG	
E132	FE3C	0358	CP	'<'	; RECURSIVE CALL
E134	200B	0359	JR	NZ,PC3	; ON CMND?
E136	F1	0360	POP	AF	
E137	3E0D	0361	LD	A,CR	; YES. CONVERT
E139	F5	0362	PUSH	AF	; '<' TO A CR.
E13A	D5	0363	PUSH	DE	
E13B	C5	0364	PUSH	BC	
E13C	CDC2E0	0365	CALL	CMND1	
E13F	C1	0366	POP	BC	
E140	D1	0367	POP	DE	
E141	F1	0368 PC3:	POP	AF	
E142	E1	0369	POP	HL	
E143	C9	0370	RET		
		0371 ;			
		0372 ;			
		0373 ; GET CHARACTER. RETURNS IT IN A.			
		0374 ; ALTERS F.			
		0375 ;			
E144	CD16E0	0376 GCHR:	CALL	GBYTE	
E147	CD12E1	0377	CALL	PCHR	
E14A	28F8	0378	JR	Z,GCHR	; IF NULL DON'T RETURN
E14C	C9	0379	RET		
		0380 ;			
		0381 ;			
		0382 ; CRLF. ALTERS A ONLY.			
		0383 ;			
E14D	3E0D	0384 CRLF:	LD	A,CR	
E14F	18C1	0385	JR	PCHR	
		0386 ;			
		0387 ;			
		0388 ; LOADS HL WITH SOURCE ADDR, BC & DE			
		0389 ; WITH THE INCREMENT. ENDS WITH A CRLF.			
		0390 ;			
E151	97	0391 L2NCR0:	SUB	A	
		0392 ;			
E152	CD8BE1	0393 L2NCR:	CALL	LD2N	
		0394 ;			
		0395 ; SKIP INITIAL SPACES.			
		0396 ; IF DELIMITER NOT A CR, ERROR			
		0397 ;			
E155	CDDEE1	0398 SKSGCR:	CALL	SKSG	; WAIT FOR NON-SPACE
E158	209B	0399	JR	NZ,ERRV1	; IF NOT CR, ERROR
E15A	EB	0400	EX	DE,HL	

```
E15B C9          0401      RET
                  0402 ;
                  0403 ;
                  0404 ; PRINT THE NUMBER IN HL, FOLLOWED BY A COLON.
                  0405 ; PRESERVES ALL REGISTERS EXCEPT A.
                  0406 ;
E15C CD4DE1      0407 PCADDR: CALL    CRLF
                  0408 ;
E15F CDF2E1      0409 PADDR:  CALL    PNHL
E162 3E3A         0410 LD     A,':'
E164 18AC         0411 JR     PCHR
                  0412 ;
                  0413 ;
                  0414 ; COMMAND
                  0415 ;
E166 CDA5E1      0416 VERIF:  CALL    L3NCR           ;GET 3 OPERANDS
                  0417 ;
                  0418 ; COMPARES TWO AREAS OF MEMORY. ENTER WITH
                  0419 ; SOURCE IN HL, DESTINATION IN DE & COUNT
                  0420 ; IN BC. ALTERS ALL REGISTERS.
                  0421 ;
E169 1A          0422 VRFY:   LD     A,(DE)
E16A EDA1         0423 CPI      ;COMPARE TO SOURCE
E16C 2B          0424 DEC     HL
E16D C4F2E1      0425 CALL    NZ,PNHL          ;PRINT SOURCE ADDR
E170 C4E9E1      0426 CALL    NZ,PSNM          ;& CONTENTS
E173 EB          0427 EX     DE,HL
E174 C4E9E1      0428 CALL    NZ,PSNM          ;& DEST CONTENTS
E177 C4EFE1      0429 CALL    NZ,PSNHL          ;& DEST ADDR
E17A C44DE1      0430 CALL    NZ,CRLF
E17D EB          0431 EX     DE,HL
E17E 23          0432 INC    HL
E17F 13          0433 INC    DE
E180 E0          0434 RET    PO                ;IF BC=0, DONE.
E181 18E6         0435 JR     VRFY
                  0436 ;
                  0437 ;
                  0438 ; COMMAND
                  0439 ;
E183 CDA5E1      0440 MOVE:   CALL    L3NCR          ;OPERANDS
E186 CD1AE2      0441 CALL    MVE               ;MOVE IT
E189 18DE         0442 JR     VRFY
                  0443 ;
                  0444 ;
                  0445 ;
                  0446 ; LOAD TWO NUMBERS. LOADS DE WITH THE BEGINNING
                  0447 ; ADDR, N1. LOADS BC & HL WITH THE INCREMENT
                  0448 ; N2-N1+1 (OR WITH N2 IF THE OPR IS 'S').
                  0449 ; RETURNS WITH LAST DELIMITER IN A.
                  0450 ;
                  0451 ;
E18B CDAEE1      0452 LD2N:   CALL    GNHL          ;N1 TO HL, DELIM TO A
E18E EB          0453 EX     DE,HL          ;SAVE N1 IN DE
E18F CDDEE1      0454 CALL    SKSG          ;GET NEXT NON-SPACE
E192 FE53         0455 CP     'S'+CASE        ;SWATH?
E194 2005         0456 JR     NZ,L2N1
                  0457 ;
```

```
E196 CDADEL      0458      CALL    GNHL0          ; YES. INCREMENT TO HL.
E199 1807         0459      JR     L2N2
0460 ;
E19B CDAEE1      0461 L2N1:   CALL    GNHL          ; INCREMENT
E19E B7           0462      OR     A              ; CLEAR CY
E19F ED52         0463      SBC   HL,DE        ; N2-N1
E1A1 23           0464      INC   HL              ; INCLUDE END POINT
E1A2 44           0465 L2N2:   LD     B,H            ; BC GETS THE INCRM
E1A3 4D           0466      LD     C,L            ; BC GETS THE INCRM
E1A4 C9           0467      RET
0468 ;
0469 ;
0470 ; LOAD 3 OPERANDS. HL GETS THE SOURCE, BC
0471 ; THE INCREMENT, AND DE THE 3RD OPERAND.
0472 ;
E1A5 CD8BEL      0473 L3NCR:  CALL    LD2N
0474 ; (CONTINUE BELOW)
0475 ;
0476 ;
0477 ; ENTER WITH SPACE OR THE FIRST DIGIT
0478 ; OF A NUMBER IN A. LOADS HL WITH
0479 ; WITH A NEW NUMBER & THEN EXCHANGES
0480 ; DE & HL. FINISHES WITH A CRLF.
0481 ;
E1A8 CDAEE1      0482 L1NCR:  CALL    GNHL          ; SKIP SPACES, LOAD HL
E1AB 18A8         0483      JR     SKSGCR        ; WAIT FOR A CR
0484 ;
0485 ;
0486 ; CLEARS HL. IF ENTERED WITH HEX CHAR IN A,
0487 ; SHIFTS IT INTO HL. O/W, IGNORES LEADING
0488 ; SPACES. FIRST CHAR MUST BE HEX. CONTINUES
0489 ; SHIFT UNTIL A NON-HEX CHAR RECEIVED & THEN
0490 ; RETURNS WITH THE LATTER IN A.
0491 ; PRESERVES B,C,D,E.
0492 ;
0493 ;
E1AD 97           0494 GNHL0:  SUB    A
0495 ;
E1AE C5           0496 GNHL:   PUSH   BC              ; SAVE
E1AF 210000        0497 LD      HL,0            ; CLR BUFFER
0498 ; STRIP LEADING SPACES & GET CHAR
E1B2 CDDEE1      0499 CALL    SKSG
0500 ; FIRST CHAR MUST BE HEX
E1B5 CDC6E1      0501 CALL    HEXSH          ; IF HEX, SHIFT INTO HL
E1B8 DAE5E0      0502 JP     C,ERROR        ; O/W, ERROR
E1BB CD44E1      0503 GN1:   CALL    GCHR
E1BE CDC6E1      0504 CALL    HEXSH          ; IF HEX SHIFT INTO HL
E1C1 78           0505 LD     A,B            ; RESTORE CHAR
E1C2 30F7         0506 JR     NC,GN1        ; IF HEX, CONTINUE
E1C4 C1           0507 POP    BC              ; IF NON-HEX, DONE
E1C5 C9           0508 RET
0509 ;
0510 ;
0511 ; IF A CONTAINS HEX CHAR, SHIFTS BINARY EQUIVALENT
0512 ; INTO HL. IF NOT HEX, RET WITH CY SET. SAVES
0513 ; ORIGINAL CHAR IN B
0514 ;
```

```
E1C6 47      0515 HEXSH: LD     B,A
E1C7 D630    0516 SUB    '0'          ; < '0'?
E1C9 D8      0517 RET     C
E1CA C6E9    0518 ADD    '0'-['G'+CASE]
E1CC D8      0519 RET     C
E1CD D6FA    0520 SUB    'A'-'G'
E1CF 3003    0521 JR     NC,HX1      ;OK IF >= 'A'
E1D1 C607    0522 ADD    ['A'+CASE]-['9']+1]
E1D3 D8      0523 RET     C
E1D4 C60A    0524 HX1:   ADD    '9'+1-'0'
0525 ; THE A-REG NOW CONTAINS THE HEX DIGIT IN BINARY.
0526 ; (THE HIGH-ORDER NIBBLE OF A IS 0.)
E1D6 29      0527 HXSH4: ADD    HL,HL      ;SHIFT 4 BITS INTO HL
E1D7 29      0528 ADD    HL,HL
E1D8 29      0529 ADD    HL,HL
E1D9 29      0530 ADD    HL,HL
E1DA B5      0531 OR     L
E1DB 6F      0532 LD     L,A
E1DC C9      0533 RET
0534 ;
0535 ;
0536 ; RETURNS WITH A NON-SPACE IN THE A-REG.
0537 ; IF ENTERED WITH A-REG CONTAINING A NULL
0538 ; OR A SPACE, GETS NEW CHARS UNTIL FIRST
0539 ; NON-SPACE OCCURS. ALTERS AF.
0540 ;
E1DD 97      0541 SKSG0: SUB    A
0542 ;
E1DE B7      0543 SKSG:  OR     A          ;DOES A CONTAIN NULL?
E1DF CC44E1  0544 SK1:   CALL   Z,GCHR
E1E2 FE20    0545 CP     20H      ;SPACE?
E1E4 28F9    0546 JR     Z,SK1
E1E6 FE0D    0547 CP     CR
E1E8 C9      0548 RET
0549 ;
0550 ;
0551 ;
0552 ; PRINT SPACE FOLLOWED BY THE NUMBER POINTED
0553 ; TO BY HL. ALTERS A ONLY.
0554 ;
E1E9 CD10E1  0555 PSNM:  CALL   SPACE
0556 ; (CONTINUE BELOW)
0557 ;
0558 ; PRINTS THE NUMBER POINTED TO BY HL.
0559 ; PRESERVES ALL REGISTERS BUT A.
0560 ;
E1EC 7E      0561 PNM:   LD     A,(HL)
E1ED 1808    0562 JR     P2HEX
0563 ;
0564 ;
0565 ;
0566 ; PRINT THE NUMBER IN HL.
0567 ; PRESERVES ALL BUT A.
0568 ;
E1EF CD10E1  0569 PSNHL: CALL   SPACE
0570 ;
E1F2 7C      0571 PNHL: LD     A,H
```

```
E1F3 CDF7E1      0572      CALL     P2HEX
E1F6 7D          0573      LD       A,L
0574 ;           ; (CONTINUE BELOW)
0575 ;
0576 ; PRINT THE NUMBER IN THE A-REGISTER.
0577 ; PRESERVES ALL REGISTERS.
0578 ;
E1F7 CDFBE1      0579 P2HEX:  CALL     P1HEX
E1FA 1F          0580      RRA
E1FB 1F          0581 P1HEX:  RRA
E1FC 1F          0582      RRA
E1FD 1F          0583      RRA
E1FE 1F          0584      RRA
E1FF F5          0585      PUSH    AF
E200 E60F         0586      AND     0FH      ; MASK
E202 FE0A         0587      CP      10D      ; <= 9?
E204 3802         0588      JR      C,PH1
E206 C607         0589      ADD     7       ;A THRU F
E208 C630         0590 PH1:   ADD     30H      ;ASCII BIAS
E20A CD12E1      0591      CALL    PCHR    ;PRINT IT
E20D F1          0592      POP     AF
E20E C9          0593      RET
0594 ;
0595 ;
0596 ; PRINT MESSAGE. ENTER WITH ADDR OF MSG
0597 ; IN HL. THE MESSAGE IS TERMINATED
0598 ; AFTER PRINTING A CHARACTER WHOSE
0599 ; PARITY BIT WAS SET.
0600 ; PRESERVES FLAGS, INCREMENTS HL.
0601 ;
0602 ;
0603 ;
E20F F5          0604 PMSG:  PUSH    AF      ;SAVE
E210 7E          0605 PS1:   LD      A,(HL)
E211 23          0606      INC     HL
E212 CD12E1      0607      CALL    PCHR
E215 17          0608      RLA
E216 30F8         0609      JR      NC,PS1  ;LAST CHARACTER?
E218 F1          0610      POP     AF      ;IF NOT, LOOP
E219 C9          0611      RET
0612 ;
0613 ;
0614 ; MOVE FROM ONE LOCATION TO ANOTHER. ENTER
0615 ; WITH SOURCE ADDR IN HL, DEST IN DE, BYTE
0616 ; COUNT IN BC. PRESERVES ALL REGISTERS.
0617 ;
E21A E5          0618 MVE:   PUSH    HL      ;SOURCE
E21B D5          0619      PUSH    DE      ;DEST
E21C C5          0620      PUSH    BC      ;BYTE COUNT
E21D EDB0         0621      LDIR
E21F C1          0622      POP     BC
E220 D1          0623      POP     DE
E221 E1          0624      POP     HL
E222 C9          0625      RET
0626 ;
0627 ;
0628 ; COMMAND
```

```
0629 ;
0630 ; GO <CR> EXECUTION BEGINS AT USER PC.
0631 ;
0632 ; COMMAND
0633 ;
0634 ; GO <ADDR1>/<ADDR2> ... <ADDRN>
0635 ; EXECUTION BEGINS AT ADDR1 WITH BREAKPOINTS SET
0636 ; AT ADDR2,...,ADDRN.
0637 ;
0638 GO:
0639 ; B GETS NBRKPT+1 (MAX. NUMBER OF BP + 1)
0640 ; C, THE BREAKPOINT FLAG, GETS 0 (NO BP SET)
E223 010006 0641 LD BC,[NBRKPT+1] SHL 8]+0
E226 CDDEE1 0642 G01: CALL SKSG ;WAIT FOR NON-SPACE
E229 283A 0643 JR Z,RETN ;RETN IF CR
E22B FE2F 0644 CP '/' ;BP?
E22D 200D 0645 JR NZ,G03
E22F 4F 0646 LD C,A ;SET BRKPT FLAG (2FH)
E230 213000 0647 LD HL,RSTLC ;TRANSFER
E233 36C3 0648 LD (HL),0C3H ;'JP SVMS' TO
E235 2145E0 0649 LD HL,SVMS
E238 223100 0650 LD (RSTLC+1),HL ;RST LOC
E23B 97 0651 SUB A
E23C CDAEE1 0652 G03: CALL GNHL ;GET ADDR
E23F CB69 0653 BIT 5,C ; FLAG SET?
E241 EB 0654 EX DE,HL
E242 DDE5 0655 PUSH IX
E244 E1 0656 POP HL
E245 2818 0657 JR Z,G05 ;JUMP IF NO BP
0658 ;
E247 05 0659 DEC B ;IF TOO MANY BP,
E248 CAE5E0 0660 JP Z,ERROR ;ERROR.
E24B 6E 0661 LD L,(HL) ;HL = BPSP
0662 ;
E24C 23 0663 INC HL ;BUMP BPSP
E24D EB 0664 EX DE,HL ;DE=BPSP, HL= BP ADDR
E24E EDA0 0665 LDI
E250 2B 0666 DEC HL
E251 36F7 0667 LD (HL),0C7H+RSTLC ;RST INSTRUCTION
E253 EB 0668 EX DE,HL ;HL=BPSP
E254 73 0669 LD (HL),E ;BP ADDR TO STACK
E255 23 0670 INC HL
E256 72 0671 LD (HL),D
E257 23 0672 INC HL
E258 360B 0673 LD (HL),BPMRK ;PUNCTUATION (BP SET)
E25A DD7500 0674 LD (IX),L
E25D 18C7 0675 JR G01
0676 ; CHANGE USER PC
E25F 2B 0677 G05: DEC HL
E260 72 0678 LD (HL),D
E261 2B 0679 DEC HL
E262 73 0680 LD (HL),E
E263 18C1 0681 JR G01 ;BACK FOR MORE
0682 ;
E265 E1 0683 RETN: POP HL ;STRIP ADDR FROM STK
E266 E1 0684 POP HL ;UHL'
E267 D1 0685 POP DE ;UDE'
```

```
E268 C1          0686      POP      BC           ;UBC'
E269 F1          0687      POP      AF           ;UAF'
E26A D9          0688      EXX
E26B 08          0689      EX       AF,AF'
0690 ;
E26C F1          0691      POP      AF           ;UIN
E26D ED47        0692      LD       I,A          ; UI
E26F F3          0693      DI
E270 3001        0694      JR      NC,RT1
E272 FB          0695      EI
0696 ;IFF NOW RESTORED
E273 FDE1        0697 RT1:    POP      IY           ;UIY
E275 DDE1        0698      POP      IX           ;UIX
E277 D1          0699      POP      DE           ;USP
0700 ;
0701 ; COPY THE REMAINDER OF THE SYS STACK
0702 ; TO THE USER STACK. IF THIS TRANSFER
0703 ; IS MADE WITHOUT ERROR, SWITCH TO THE
0704 ; USER STACK. OTHERWISE, RETAIN THE
0705 ; SYSTEM STACK.
0706 ;
E278 210A00      0707      LD       HL,10D
E27B 45          0708      LD       B,L
E27C 39          0709      ADD     HL,SP
E27D EB          0710      EX       DE,HL
E27E 1B          0711 RT2:    DEC     DE
E27F 2B          0712      DEC     HL
E280 1A          0713      LD       A,(DE)
E281 77          0714      LD       (HL),A
E282 BE          0715      CP      (HL)
E283 2003        0716      JR      NZ,RT3
E285 10F7        0717      DJNZ   RT2
E287 F9          0718      LD       SP,HL
0719 ;
E288 E1          0720 RT3:    POP      HL
E289 D1          0721      POP      DE
E28A C1          0722      POP      BC
E28B F1          0723      POP      AF
E28C C9          0724      RET
0725 ;
0726 ;
0727 ; COMMAND. DISPLAY REGISTERS.
0728 ;
0729 ; DR
0730 ;
0731 ; COMMAND. DISPLAY MEMORY.
0732 ;
0733 ; DM <STARTING ADDR> <ENDING ADDR OR SWATH>
0734 ;
0735 ;
E28D 018041      0736 DISPL:   LD       BC,[[ 'A'+CASE] SHL 8]+80H ;[FOR DR}
E290 203F        0737      JR      NZ,SUBR2      ;IF NOT 'M', DR
0738 ;
0739 ;
E292 CD51E1      0740 DSPM:   CALL    L2NCR0      ;GET OPERANDS
E295 1610        0741 DSPM1:  LD       D,16          ;BYTE COUNT
E297 CD5CE1      0742      CALL    PCADDR      ;ADDRESS
```

```

E29A CDE9E1      0743 DM2:    CALL    PSNM      ;MEM CONTENTS
E29D EDA1        0744 CPI      ;INC HL & DEC BC
E29F E24DE1      0745 JP      PO,CRLF
E2A2 15          0746 DEC      D
E2A3 28F0        0747 JR      Z,DSPM1
E2A5 7A          0748 LD      A,D
E2A6 E603        0749 AND     3
E2A8 CC10E1      0750 CALL    Z,SPACE
E2AB CC10E1      0751 CALL    Z,SPACE
E2AE 18EA        0752 JR      DM2
0753 ;
0754 ;
0755 ; COMMAND. SUBSTITUTE MEMORY LOCATION.
0756 ;
0757 ; SM <ADDR>
0758 ;
0759 ; COMMAND. SUBSTITUTE USER-REGISTER.
0760 ;
0761 ; S<REGISTER NAME>
0762 ;
0763 ; REGISTER NAMES: P [PC], S [SP],
0764 ; A, F, B, C, D, E, H [HL],
0765 ; I, N [IFF], X [IX], Y [IY],
0766 ; A',F',B',C',D',E',H' [HL'].
0767 ;
0768 ;
E2B0 2016        0769 SUBST:  JR      NZ,SUBR   ;IN NOT 'M', SR
0770 ;
0771 ;
E2B2 97          0772 SUBM:   SUB     A
E2B3 47          0773 LD      B,A      ;1-BYTE MASK
E2B4 CDA8E1      0774 CALL    L1NCR
E2B7 EB          0775 EX      DE,HL   ;HL GETS ADDR
E2B8 CC5CE1      0776 SM1:   CALL    Z,PCADDR
E2BB CC10E1      0777 CALL    Z,SPACE
0778 ; PRINT CURRENT VALUE, REQUEST NEW VALUE &
0779 ; PRINT IT IF GIVEN
E2BE CD32E3      0780 CALL    GSUBV
E2C1 C8          0781 RET     Z       ;IF CR, DONE.
E2C2 23          0782 INC     HL
E2C3 3E07        0783 LD      A,7      ;PRINT ADDRESS IF IT
E2C5 A5          0784 AND     L       ;IS A MULTIPLE OF 8
E2C6 18F0        0785 JR      SM1
0786 ;
0787 ;
E2C8 47          0788 SUBR:  LD      B,A
E2C9 CD44E1      0789 CALL    GCHR
E2CC FE27        0790 CP      ''
E2CE 2002        0791 JR      NZ,SR2
E2D0 0C          0792 INC     C      ;TURN ON THE PRIME-FLAG
E2D1 97          0793 SUBR2: SUB     A
E2D2 CD55E1      0794 SR2:   CALL    SKSGCR  ;WAIT FOR CR
E2D5 78          0795 SR3:   LD      A,B
E2D6 D641        0796 SUB     'A'+CASE ;CHECK THE RANGE
E2D8 DAE5E0      0797 JP      C,ERROR
E2DB FE19        0798 CP      'Y'-'A'+1
E2DD D2E5E0      0799 JP      NC,ERROR

```

E2E0	5F	0800	LD	E,A	
E2E1	1600	0801	LD	D,0	
E2E3	21D7E3	0802	LD	HL,RGTBL	
E2E6	19	0803	ADD	HL,DE	
E2E7	7E	0804	LD	A,(HL)	
E2E8	B7	0805	OR	A	
E2E9	2833	0806	JR	Z,SR6	; IF ENTRY = 0, SKIP
E2EB	1E00	0807	LD	E,0	
E2ED	CB41	0808	BIT	0,C	; PRIME?
E2EF	2806	0809	JR	Z,SR4	
E2F1	CB76	0810	BIT	PF,(HL)	; YES. PRIMEABLE REG?
E2F3	2829	0811	JR	Z,SR6	; IF NOT, SKIP.
E2F5	1E10	0812	LD	E,DUAFF-DUAFF2	
E2F7	E61F	0813 SR4:	AND	1FH	; STRIP FLAGS FROM ENTRY
E2F9	83	0814	ADD	E	
E2FA	5F	0815	LD	E,A	
E2FB	C5	0816	PUSH	BC	; SAVE
E2FC	78	0817	LD	A,B	; PRINT REG NAME
E2FD	CD12E1	0818	CALL	PCHR	
E300	FE48	0819	CP	'H'+CASE	
E302	3E4C	0820	LD	A,'L'+CASE	
E304	CC12E1	0821	CALL	Z,PCHR	
E307	EE71	0822	XOR	'L'+CASE XOR '=';CLEAR CY, A = '='.	
E309	CB41	0823	BIT	0,C	; PRIME?
E30B	2802	0824	JR	Z,SR5	
E30D	3E27	0825	LD	A,''''	
E30F	CD12E1	0826 SR5:	CALL	PCHR	
E312	46	0827	LD	B,(HL)	; SAVE ORIGINAL ENTRY
E313	DDE5	0828	PUSH	IX	
E315	E1	0829	POP	HL	; STACK FRAME
E316	ED52	0830	SBC	HL,DE	; HL -> USER REG
E318	CD32E3	0831	CALL	GSUBV	; PRINT VALUE, REQUEST NEW
E31B	78	0832	LD	A,B	; SAVE
E31C	C1	0833	POP	BC	
E31D	C8	0834	RET	Z	; DONE IF CR
		0835 ;			
E31E	04	0836 SR6:	INC	B	; NEXT REG
E31F	07	0837	RLCA		; Y OR H?
E320	30B3	0838	JR	NC,SR3	; IF NEITHER, LOOP
E322	07	0839	RLCA		; YES, IS IT Y?
E323	CD4DE1	0840 SUBR3:	CALL	CRLF	; [ENTRY FOR DISPLAYING PC
E326	3805	0841	JR	C,SR8	
E328	0641	0842	LD	B,'A'+CASE	; YES, IT IS Y.
E32A	0C	0843	INC	C	; TURN ON PRIME-FLAG
E32B	18A8	0844	JR	SR3	
E32D	CB41	0845 SR8:	BIT	0,C	; NO. H OR H'?
E32F	28A4	0846	JR	Z,SR3	; IF H, LOOP.
E331	C9	0847	RET		; IT IS H'. DONE.
		0848 ;			
		0849 ;			
		0850 ; ENTER WITH HL POINTING TO MEMORY &			
		0851 ; B CONTAINING THE 1-BYTE OR 2-BYTE FLAG.			
		0852 ; PRINTS SPACE, CONTENTS OF (HL), & ALSO (HL-1) FOR			
		0853 ; 2-BYTE REGS, GETS SUBSTITUTION VALUE & LOADS IT.			
		0854 ; RETURNS WITH Z-FLAG SET IFF THE DELIMITER IS			
		0855 ; A CARRIAGE-RETURN.			
		0856 ; PRESERVES BC & HL.			

```
0857 ;
E332 CDECE1 0858 GSUBV: CALL PNM ;PRINT (HL)
E335 CB68 0859 BIT B2F,B ;2-BYTE REG?
E337 2804 0860 JR Z,GS1
E339 2B 0861 DEC HL
E33A CDECE1 0862 CALL PNM ;LO BYTE
E33D 79 0863 GS1: LD A,C ;SUBST-OR-DISPLAY FLAG
E33E 07 0864 RLCA
E33F 380A 0865 JR C,GS2 ;IF DISPLAY, EXIT.
E341 3E2E 0866 LD A,'.'
E343 CD12E1 0867 CALL PCHR
E346 CD44E1 0868 CALL GCHR
E349 FE2F 0869 CP '.'+1 ;SUBSTITUTION?
E34B DC12E1 0870 GS2: CALL C,PCHR ;IF NOT, PRINT ANOTHER.
E34E 380C 0871 JR C,GS3
E350 EB 0872 EX DE,HL
E351 CDAEE1 0873 CALL GNHL ;NEW VALUE
E354 EB 0874 EX DE,HL
E355 73 0875 LD (HL),E
E356 CB68 0876 BIT B2F,B
E358 2802 0877 JR Z,GS3
E35A 23 0878 INC HL
E35B 72 0879 LD (HL),D
E35C FE0D 0880 GS3: CP CR
E35E C410E1 0881 CALL NZ,SPACE
E361 C9 0882 RET
0883 ;
0884 ;
0885 ;...SUBDM 00 7E 5 585 BY 5 100 DBE++
0886 ;
0887 ;
0888 ; COMMAND
0889 ; SELECT UART-A OR UART-B.
0890 ;
0891 ; UA
0892 ; UB
0893 ;
E362 CDA8E1 0894 UART: CALL L1NCR ;A OR B?
E365 7B 0895 LD A,E
E366 FE0B 0896 CP 0BH
E368 2005 0897 JR NZ,UARTA
E36A 3E80 0898 LD A,80H
E36C D304 0899 OUT APARLP,A
E36E C9 0900 RET
0901 ;
E36F 97 0902 UARTA: SUB A
E370 D354 0903 OUT BPAPRLP,A
E372 C9 0904 RET
0905 ;
0906 ;
0907 ; COMMAND
0908 ; READ BINARY INPUT FROM DATA PORT
0909 ;
E373 CD52E1 0910 READB: CALL L2NCR ;GET MEM ADDRS
E376 CD0EE0 0911 RB1: CALL CHKIN ;GET INPUT
E379 28FB 0912 JR Z,RB1
E37B 77 0913 LD (HL),A ;TO MEM
```

```
E37C EDA1          0914      CPI
E37E EØ            0915      RET      PO
E37F 18F5          0916      JR       RB1
0917 ;
0918 ;
0919 ; COMMAND
0920 ; WRITE BINARY OUTPUT TO DATA PORT
0921 ;
E381 CD52E1        0922 WRITB: CALL    L2NCR      ;GET MEM ADDRS
E384 7E             0923 WBl:   LD     A,(HL)
E385 CD1EEØ         0924      CALL    PBYTE
E388 EDA1          0925      CPI
E38A EØ            0926      RET     PO
E38B 18F7          0927      JR     WB1
0928 ;
0929 ;
0930 ; COMMAND
0931 ; PRINT NULLS ON THE CURRENT DEVICE.
0932 ;
0933 ; N <NUMBER-OF-NULLS>
0934 ;
E38D CDA8E1        0935 NULLS: CALL    L1NCR
E390 43             0936      LD     B,E
E391 97             0937      SUB    A
E392 CD12E1        0938 N2:   CALL    PCHR
E395 10FB           0939      DJNZ   N2
E397 C9             0940      RET
0941 ;
0942 ;
0943 ; COMMAND
0944 ; OUT <DATA-BYTE> <PORT NNUMBER>
0945 ;
E398 CDAEEL        0946 OUTP:  CALL    GNHL
E39B EB             0947      EX     DE,HL      ;E GETS DATA
E39C CDA8E1        0948      CALL    L1NCR      ;GET PORT NUMBER
0949 ;
E39F 4B             0950      LD     C,E      ; TO C
E3A0 ED69           0951      OUT    (C),L
E3A2 C9             0952      RET
0953 ;
0954 ;
0955 ; BAUD RATES.
0956 ; WITH THE CROMEMCO TUART: 19200, 9600, 4800,
0957 ;                               2400, 1200, 300, 150, 110.
0958 ;
0959 ; WITH THE 3P+S: 2400, 300, 110.
0960 ;
0961 ;
E3A3 94CEA292      0962 BAUDRS: DB      94H,0CEH,0A2H,92H,88H,84H,82H,1
     88848201
0963 ;
0964 ;
E3AB 0A0080         0965 LFNN:  DB      LF,0,0 OR 80H
0966 ;
0967 ;
E3AE BA             0968 PRMPT: DB      ':' OR 80H
0969 ; THE COMMAND TBL MUST IMMEDIATELY FOLLOW
```

0970 ; THE PROMPT MESSAGE				
E3AF	8DE2	0971	DW	DISPL ;DISPLAY: DM, DR
E3B1	E5E0	0972	DW	ERROR ;E
E3B3	E5E0	0973	DW	ERROR ;F
E3B5	23E2	0974	DW	GO ;GO; GO/WITH BREAKPOINTS
E3B7	E5E0	0975	DW	ERROR ;H
E3B9	2DE0	0976	DW	INITBAUD ;INITIALIZE BAUD RATE
E3BB	E5E0	0977	DW	ERROR ;J
E3BD	E5E0	0978	DW	ERROR ;K
E3BF	E5E0	0979	DW	ERROR ;L
E3C1	83E1	0980	DW	MOVE ;MOVE A BLOCK OF MEMORY
E3C3	8DE3	0981	DW	NULLS ;NULLS
E3C5	98E3	0982	DW	OUTP ;OUTPUT
E3C7	ECE0	0983	DW	PROG ;PROGRAM
E3C9	E5E0	0984	DW	ERROR ;Q
E3CB	73E3	0985	DW	READB ;READ BINARY OR ASCII
E3CD	B0E2	0986	DW	SUBST ;SUBSTITUTE: SM, SA, SB,
E3CF	E5E0	0987	DW	ERROR ;T
E3D1	62E3	0988	DW	UART ;UART: UA, UB
E3D3	66E1	0989	DW	VERIF ;VERIFY BLOCKS OF MEMORY
E3D5	81E3	0990	DW	WRITB ;WRITE BINARY OR ASCII
		0991 ;		
(0040)		0992 PM:	EQU	1 SHL PF ;PRIMEABLE-REG MASK
(0000)		0993 B1M:	EQU	0 ;1-BYTE REG MASK
(0020)		0994 B2M:	EQU	1 SHL B2F ;2-BYTE REG MSK
(0080)		0995 CRM:	EQU	1 SHL CRF ;CARRIAGE-RETURN MSK
		0996 ;		
E3D7	43	0997 RGTBL:	DB	-DUAF OR PM ;A
E3D8	45	0998	DB	-DUBC OR PM ;B
E3D9	46	0999	DB	-DUBC+1 OR PM ;C
E3DA	47	1000	DB	-DUDE OR PM ;D
E3DB	48	1001	DB	-DUDE+1 OR PM ;E
E3DC	44	1002	DB	-DUAF+1 OR PM ;F
E3DD	00	1003	DB	0
E3DE	E9	1004	DB	-DUHL OR PM OR B2M OR CRM ;H [HL]
E3DF	11	1005	DB	-DUIN OR B1M ;I
E3E0	00	1006	DB	0
E3E1	00	1007	DB	0
E3E2	00	1008	DB	0
E3E3	00	1009	DB	0
E3E4	12	1010	DB	-DUIN+1 OR B1M ;N [INTERRUPT FF]
E3E5	00	1011	DB	0
E3E6	21	1012	DB	-DUPC OR B2M ;PC
E3E7	00	1013	DB	0
E3E8	00	1014	DB	0
E3E9	2B	1015	DB	-DUSP OR B2M ;SP
E3EA	00	1016	DB	0
E3EB	00	1017	DB	0
E3EC	00	1018	DB	0
E3ED	00	1019	DB	0
E3EE	2D	1020	DB	-DUIX OR B2M ;X [IX]
E3EF	AF	1021	DB	-DUIY OR B2M OR CRM ;Y [IY]
		1022 ;		
		1023 ;		
E3F0	0D0D4352	1024 HEAD:	DB	CR,CR,'CROMEMCO ZML.', '4' OR 80H
	4F4D454D			
	434F205A			

4D312EB4

1025 ;

Errors	0
Range Count	0

## Symbol Table

ABAUDP	0000	ACMNDP	0002	ALT	007D	APARLP	0004	B1M	0000
B2F	0005	B2M	0020	BAUDRS	E3A3	BCMNDP	0052	BPARLP	0054
BPMRK	000B	BPSTOR	0016	CASE	0000	CHKIN	E00E	CL1	E0A4
CL2	E0B3	CLBP	E0A0	CMND	E0BE	CMND1	E0C2	COM1	E051
COM3	E082	COM4	E08B	COMMON	E04A	CR	000D	CRF	0007
CRLF	E14D	CRM	0080	CSTART	E000	DATA	0001	DAV	0040
DISPL	E28D	DM2	E29A	DSPM	E292	DSPM1	E295	DUAF	FFF4
DUAF2	FFED	DUBC	FFF8	DUBC2	FFEB	DUDE	FFF9	DUDE2	FFE9
DUHL	FFF7	DUHL2	FFE7	DUIN	FFEF	DUIX	FFF3	DUIY	FFF1
DUPC	FFFF	DUSP	FFF5	ERROR	E0E5	ERRV1	E0F5	ESC	001B
ESCPE	E0EA	GBYTE	E016	GCHR	E144	GN1	E1BB	GNHL	E1AE
GNHLØ	E1AD	GO	E223	G01	E226	GO3	E23C	GO5	E25F
GS1	E33D	GS2	E34B	GS3	E35C	GSUBV	E332	HEAD	E3F0
HEXSH	E1C6	HX1	E1D4	HXSH4	E1D6	INIT	E029	INITBA	E02D
IT1	E034	L1NCR	E1A8	L2N1	E19B	L2N2	E1A2	L2NCR	E152
L2NCRØ	E151	L3NCR	E1A5	LD2N	E18B	LENRGS	001A	LF	000A
LFNN	E3AB	MOVE	E183	MVE	E21A	N2	E392	NBRKPT	0005
NULLS	E38D	OUTP	E398	P1HEX	E1FB	P2HEX	E1F7	P2NMS	E108
PADDR	E15F	PBY1	E01F	PBYTE	E01E	PC1	E113	PC2	E122
PC3	E141	PCADDR	E15C	PCHR	E112	PF	0006	PH1	E208
PM	0040	PMSG	E20F	PNHL	E1F2	PNM	E1EC	PR1	E0FB
PRMPT	E3AE	PROG	E0EC	PS1	E210	PSNHL	E1EF	PSNM	E1E9
RB1	E376	READB	E373	RETN	E265	RGTBL	E3D7	RSTLC	0030
RT1	E273	RT2	E27E	RT3	E288	SK1	E1DF	SKSG	E1DE
SKSGØ	E1DD	SKSGCR	E155	SM1	E2B8	SPACE	E110	SR2	E2D2
SR3	E2D5	SR4	E2F7	SR5	E30F	SR6	E31E	SR8	E32D
STAT	0000	SUBM	E2B2	SUBR	E2C8	SUBR2	E2D1	SUBR3	E323
SUBST	E2B0	SVMS	E045	TBE	0080	TEMPS	0016	UART	E362
UARTA	E36F	VERIF	E166	VRFY	E169	WB1	E384	WRITB	E381
WSTART	E008								

**Cross Reference**

ABAUDP	0007	0131
ACMNDP	0006	0133
ALT	0028	0345
APARLP	0008	0899
B1M	0993	1005 1010
B2F	0021	0859 0876 0994
B2M	0994	1004 1012 1015 1020 1021
BAUDRS	0962	0130
BCMNDP	0009	0111
BPARLP	0010	0110 0903
BPMRK	0017	0233 0673
BPSTOR	0015	0016
CASE	0020	0221 0269 0285 0455 0518 0522 0736 0796 0819 0820 0822 0842
CHKIN	0081	0090 0347 0911
CL1	0232	0243
CL2	0245	0234
CLBP	0228	0294
CMND	0258	0258
CMND1	0260	0365
COM1	0164	0168
COM3	0203	0194
COM4	0208	0206
COMMON	0154	0065 0076
CR	0025	0137 0356 0361 0384 0547 0880 1024 1024
CRF	0023	0995
CRLF	0384	0257 0407 0430 0745 0840
CRM	0995	1004 1021
CSTART	0061	
DATA	0005	0084 0103
DAV	0011	0082
DISPL	0736	0971
DM2	0743	0752
DSPM	0740	
DSPM1	0741	0747
DUAF	0034	0812 0997 1002
DUAF2	0042	0812
DUBC	0035	0998 0999
DUBC2	0043	
DUDE	0036	1000 1001
DUDE2	0044	
DUHL	0037	0182 1004
DUHL2	0045	0047
DUIN	0041	1005 1010
DUIX	0039	1020
DUIY	0040	1021
DUPC	0033	0047 0182 1012
DUSP	0038	1015
ERROR	0292	0270 0272 0308 0502 0660 0797 0799 0972 0973 0975 0977 0978 0979 0984 0987
ERRV1	0308	0399
ESC	0027	0343
ESCPE	0294	0344 0346
GBYTE	0090	0091 0135 0136 0376
GCHR	0376	0378 0503 0544 0789 0868
GN1	0503	0506
GNHL	0496	0452 0461 0482 0652 0873 0946
GNHL0	0494	0458

GO	0638	0974
GO1	0642	0675 0681
GO3	0652	0645
GO5	0677	0657
GS1	0863	0860
GS2	0870	0865
GS3	0880	0871 0877
GSUBV	0858	0780 0831
HEAD	1024	0219
HEXSH	0515	0501 0504
HX1	0524	0521
HXSH4	0527	
INIT	0110	0201
INITBA	0130	0976
IT1	0133	0139
L1NCR	0482	0774 0894 0935 0948
L2N1	0461	0456
L2N2	0465	0459
L2NCR	0393	0910 0922
L2NCR0	0391	0740
L3NCR	0473	0302 0416 0440
LD2N	0452	0393 0473
LENRGS	0047	0249
LF	0026	0965
LFNN	0965	0355
MOVE	0440	0980
MVE	0618	0313 0441
N2	0938	0939
NBRKPT	0014	0015 0641
NULLS	0935	0981
OUTP	0946	0982
P1HEX	0581	0579
P2HEX	0579	0562 0572
P2NMS	0327	
PADDR	0409	
PBY1	0099	0101
PBYTE	0098	0354 0924
PC1	0342	0348
PC2	0350	
PC3	0368	0359
PCADDR	0407	0742 0776
PCHR	0341	0293 0377 0385 0411 0591 0607 0818 0821 0826 0867 0870 0938
PF	0022	0810 0992
PH1	0590	0588
PM	0992	0997 0998 0999 1000 1001 1002 1004
PMMSG	0604	0220 0261 0357
PNHL	0571	0409 0425
PNM	0561	0327 0329 0858 0862
PR1	0312	0318
PRMPT	0968	0260
PROG	0302	0983
PS1	0605	0609
PSNHL	0569	0429
PSNM	0555	0426 0428 0743
RB1	0911	0912 0916
READB	0910	0985
RETN	0683	0643

RGtbl	0997	0802
Rstlc	0019	0647 0650 0667
RT1	0697	0694
RT2	0711	0717
RT3	0720	0716
SK1	0544	0546
SKSG	0543	0398 0454 0499 0642
SKSG0	0541	0267 0284
SKSGCR	0398	0483 0794
SM1	0776	0785
SPACE	0335	0555 0569 0750 0751 0777 0881
SR2	0794	0791
SR3	0795	0838 0844 0846
SR4	0813	0809
SR5	0826	0824
SR6	0836	0806 0811
SR8	0845	0841
STAT	0004	0081 0099
SUBM	0772	
SUBR	0788	0769
SUBR2	0793	0737
SUBR3	0840	0222
SUBST	0769	0986
SVMS	0146	0649
TBE	0012	0100
TEMPS	0016	0163
UART	0894	0988
UARTA	0902	0897
VERIF	0416	0989
VRFY	0422	0320 0435 0442
WB1	0923	0927
WRITB	0922	0990
WSTART	0073	